Windows Workflow Foundation

Thomas Reinwart

1. Workflow -Einführung

Warum einen Workflow verwenden

Ein Workflow ist ein Modell bestehend aus Aktivitäten das einen Prozess beschreibt. Diese Aktivitäten koordinieren Personen oder Maschinen.

Mit der Einbindung eines Workflows gibt es in der Architektur einer Applikation neue Möglichkeiten, die *Business Logic (BL)* zu definieren. Statt wie bisher unüberschaubaren Code mit vielen `if else`-Zweigen zu kodieren, gibt es nun die Möglichkeit, dies in einen Workflow zu verpacken. So lässt sich nun die BL mittels eines graphischen Designers erstellen, damit ist dies wesentlich flexibler bei Veränderungen. Zudem ist es übersichtlich dokumentiert. Das betrifft auch weit komplexere Prozesse eines Unternehmens. Mit einem Workflow lassen sich Abläufe besser koordinieren.

Windows Workflow Foundation (WF)

Der Vorteil, einen Workflow zu verwenden ist, dass damit der Prozess recherchiert, beschrieben und daraus ein Modell erstellt wird. Meist werden die Prozesse als *UML*, *use cases* oder *flow chart* dokumentiert. Beim Verwenden von WF sind die Beschreibung und der Workflow ein und das selbe System, d.h. es gibt eine visualisierte Darstellung des Workflow beim Design und zur Laufzeit, der Code der Activities ist mit dem Workflow verbunden. Somit gibt es keine getrennte Dokumentation die auseinander laufen kann.

Welcher meiner laufenden Workflows derzeit in welchem Status sich in welchem Step befindet, zeigt mir Tracking im selben Design visuell an.

Ein Workflow kann im Designer verändert werden, ohne dass dabei Code geändert werden muss. Ich kann einen Workflow in ein Produkt einbauen, dabei wird jeder Workflow für den Kunden angepasst, der Rest bleibt gleich. Bzw. der Kunde passt sich seinen Workflow selber an.

Meist läuft am Rechner eine Vielzahl von Programmen, die nur von Zeit zu Zeit echte Dienste leisten. Die meiste Zeit jedoch verbringen diese Anwendungen damit, auf Usereingaben oder auf ein anderes programatisches Event zu warten. Die Anwendungen laufen also und tun nichts anderes als Ressourcen zu verbrauchen.

In einem Workflow können die Objekte serialisiert und derserialisiert werden, die Persistierung in die Datenbank. Somit ist der Status, in dem sich der Workflow der Anwendung befindet eingefroren. Wenn nun ein bestimmtes Event eintritt, beginn der Workflow wieder zu arbeiten. Durch diese persistieren des Zustandes ist es möglich, das Laden auf einen anderen Rechner und Zeitpunkt durchzuführen. Man hat damit selber keinen Implementierungsaufwand, sich um die Persistierung eines Workflows Status zu kümmern.

Ein Workflow kann in jedem Projekttyp eingebunden werden, vom simplen Comand Line Tool bis Windows Forms, XAML, Asp.net. Innerhalb des Workflows kann über Code Activities wiederrum eigene Assemblies aufgerufen werden, oder auch WebServices eingebunden werden.

Komponenten

- *Base Activity Library*: mit der WF mitgelieferte Activities
- *Runtime engine*
- *Runtime services*: Hosting und Kommunikation
- *WF Visual Designer*: Control kann auch in der eigener Applikation gehostet werden

1.1 Sequential oder State Workflows verwenden

Die Windows Workflow Foundation bietet zwei Workflow Arten an, den *Sequential* und den *State Workflow*. Beim Projektbeginn muss ich mir im Klaren sein, welcher Workflow besser zu meinem Projekt passt.

1.1.1 Der State Workflow

Der Workflow befindet sich in einem bestimmten Status und wartet auf Events um in einen anderen Status zu gelangen. Der neue Status kann irgendein anderer Status sein. Status können verschachtelt sein. Gut geeignet für menschliche Interaktion.

Dieser Workflow führt die Aktivitäten aufgrund des Status aus. Der Workflow hängt stark von den externen Modulen ab. Userinteraktion bzw. Ergebnis der externen Module beeinflussen den Status. Zustandsübergänge werden durch externe Events ausgelöst.

Er eignet sich dann, wenn mehr Useraktion gewünscht wird. Wenn dieser Workflow eine Benachrichtigung an einen anderen Benutzer oder ein anderes System sendet, gibt es Antworten in Form von Events. Dadurch wird ein Prozess Status durch einen anderen Prozess Status getauscht.

Ein Bespiel für einen State Workflow ist das Zusammenspiel von verschiedenen Interaktionen einer Produktion eine Maschine mit einem Bediener.

Start – Material entnehmen – Material reinigen – Material bearbeiten – Material Kontrolle – Material bearbeiten – Material Ausschuss oder ***OK***

Dazwischen kann jeder einzelne Step auch wieder zum Vorgänger springen. Der Workflow wird hier lange Zeit in einem der Steps verbringen.

Der Unterschied zum *Sequential Workflow* ist, dass beim *State Workflow* auch ein vorheriger Status erreicht werden kann.

1.1.2 Der Sequential Workflow

Der Plan eines sequentiellen Workflows ist vorgegeben. Er kann Bedingungen, Schleifen usw. beinhalten. Dies passt mehr zu automatisierten Prozessen oder starren Protokollen. Er hat einen definierten Start und ein definiertes Ende, kann aber auch ewig laufen.

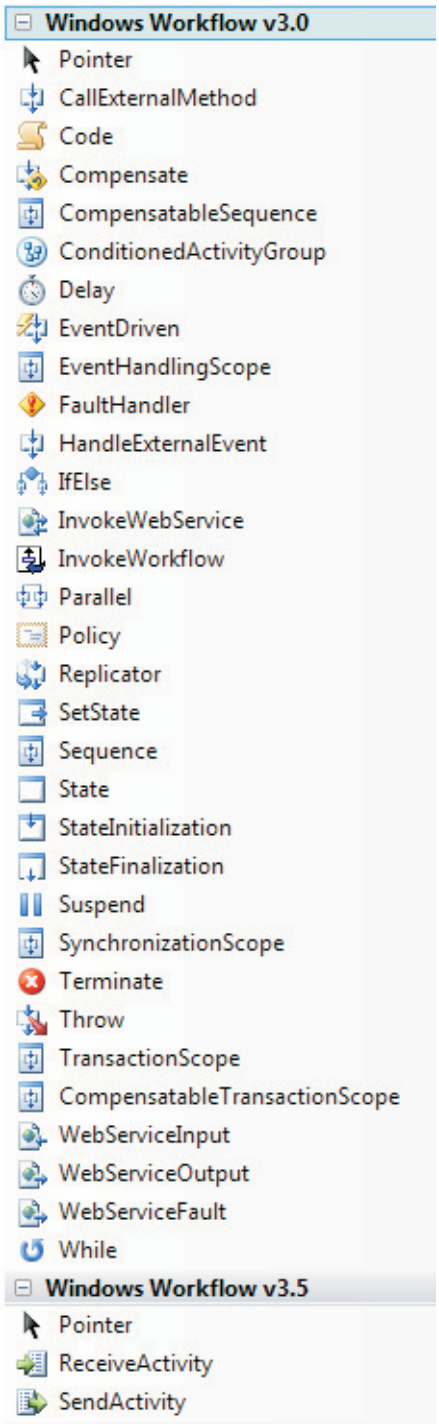
Dieser Workflow führt die Activities hintereinander aus. Der Workflow hat dabei ausführbaren Tasks unter Kontrolle. Es kann Userinteraktion geben, der Workflow kann aber auch vollautomatisch laufen.

Er ist ideal für die Umsetzung von Business Prozessen, die komplex sind.

Dieser Workflow arbeitet eine Activity nach der anderen ab, es sei denn eine Exception wird während aber Workflow Abarbeitung gefeuert oder wenn eine TerminateActivity ausgeführt wird.

Beispiel: Daten von einer Quelle lesen, Daten verarbeiten, eine Benachrichtigung schicken, Ergebnisdaten schreiben. Die Userinteraktion ist also nicht unbedingt notwendig.

Zur Verfügung stehende Activities im Workflow



Activity: alles in einem Workflow ist eine Activity, auch der Workflow selber, der einen bestimmten Typ einer Activity darstellt.

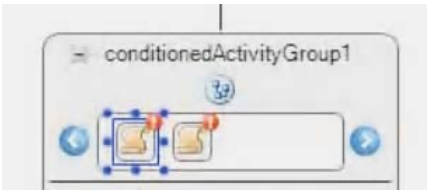
CallExternalMethod: Damit lassen sich Methoden außerhalb eines Workflows aufrufen. InterfaceType und MethodName müssen dabei definiert werden. InterfaceType gibt an, welches Runtime Service genutzt werden soll, wenn einen Activity ausgeführt wird. MethodName gibt an, welche Methode des angegebenen Interface aufgerufen werden soll.

Code: Hiermit lässt sich eigener .net Code im Workflow ausführen. Der Sourcecode ist getrennt vom Workflow.

Compensate: Ist eine Error Activity und kann zu einer Exception Handler Activity hinzugefügt werden. Kann genutzt werden um ein Rollback der Änderungen im Falle eines Fehlers durchzuführen.

Compensatable Sequence: Kann für einen bestimmten Typ eines Fehlers verwendet werden. Funktioniert ähnlich wie mit Datenbank Transaktionen. Der Unterschied ist, das die Datenbank solange sperrt bis alle Updates fertig sind, also für eine kurze Zeit. Eine Compensate Action kann bei Transaktionen verwendet werden, die lange andauern, um die original Transaktion rückgängig zu machen.

Conditioned Activity Group: Dies ist eine bedingte Aktivität, die andere Aktivitäten, basierend auf Bedingungen, die für die Gruppe gelten, oder die Aktivitäten die der Gruppe zugeordnet, ausführt.



Delay: Bei der Verwendung einer DelayActivity wird der Wartezeitraum und die Methode angegeben, die im Fall der Zeitüberschreitung ausgeführt wird. Im Beispiel eine Genehmigungsworkflows kann man ein Delay verwenden. Wenn nun eine Person innerhalb des angegeben Zeitraums nicht reagiert, wird der Alternativzweig im Workflow (z.B. andere Person kontaktieren) verwendet.

Event Driven:Diese Activity kann andere Activities beinhalten, die ausgeführt werden wenn dieses Event eintritt.

Event Handling Scope*: kann eine Anzahl von Activities beinhalten. Während diese Activities ausgeführt werden, können die Events empfangen werden.*

Fault Handler: Eine Error handling Activity, die wie ein catch block funktioniert.

Handle External Event: Die HandleExternalEventActivity wird in Verbindung mit der CallExternalMethodActivity für die In und Out Kommunikation mit einem lokalen Dienst verwendet. Sie können diese Aktivitäten direkt für die allgemeine Kommunikation verwenden. Oder Sie können Unterklassen von HandleExternalEventActivity und CallExternalMethodActivity erstellen um Aktivitäten, die streng an bestimmte Ereignisse und Methoden an ein Interface gebunden ist das ein ExternalDataExchangeAttribute besitzt, zu erzeugen.

If Else: Kann man sich als if else so wie in der Programmiersprache vorstellen. Über eine Condition wird die Bedingung festgelegt.

Invoke Web Service: kann ein Methode eines WebService mittels Proxyklasse ausführen. Parameter können übergeben und Empfangen werden.

Invoke Workflow: Ruft den zugeordneten Workflow auf Parallel: Im Workflow werden Activities als single thread ausgeführt. Es ist also keine wirkliche parallele Ausführung. Die Workflow runtime verarbeitet die Activity Queue pro Workflow Instanz im Prinzip von FIFO (first in, first out) ab.

Policy: Dies stellt eine Auflistung von Regeln dar. Eine Regel hat eine Bedingung und Aktion(en), die durchgeführt werden, wenn die Bedingung erfüllt ist. Das erlaubt einen regelbasierten Workflow, anstatt einem IfElse basierendem Workflow.

Replicator: Ist auch eine Bedingungsaktivität. Wie wenn man eine "For Each" Anweisung verwendet. Zur Laufzeit werden eine Anzahl von Instanzen einer Aktivität erstellt, die abgeschlossen werden müssen, bevor die Replikator Aktivität beendet wird.

Set State Activity: Dies ist ein flow activity, die für den Statuswechsel bei einem State Machine Workflow verwendet wird.

State Activity: Ist eine flow activity und stellt einen Status in einem State Machine Workflow dar.

State Initialization Activity: Diese Aktivität ist Bestandteil einer State Aktivität, die aus anderen Aktivitäten besteht, die ausgeführt werden, wenn die State Aktivität initialisiert wird.

Sequence: Ist eine zusammengesetzte Aktivität, die mehrere sequenzielle Aktivitäten beinhaltet. Es bietet eine einfache Möglichkeit, diese Tätigkeiten zu verknüpfen, die damit in Folge ausgeführt werden.

Suspend: Ist eine flow-Aktivität und kann den Ablauf eines Workflows für ein Einschreiten pausieren, wenn eine Fehlerbedingung (error condition) eintritt.

Synchronization Scope: Synchronisiert Workflow Aktivitäten

Terminate: Diese Aktivität beendet die Ausführung des Workflows sofort, wenn ein Fehler auftritt. Sie protokolliert auch den Fehler.

Throw: Mit dieser Aktivität können Sie eine Exception auslösen.

Transaction Scope: Diese Aktivität bietet Transaktionsunterstützung. In dieser Aktivität können sich Aktivitäten befinden, die eine Transaktion bilden.

Compensatable Transaction Scope: Es gibt zwei Aktivitäten die dieses Interface implementieren: CompensatableTransactionScopeActivity und CompensatableSequenceActivity.

WebService Input, Output, Fault: WF WebService Integration

While: Dies ist eine bedingte Aktivität und ist wie eine `"while"` Bedingung in der Anwendung. Es wird eine andere Aktivität ausgeführt, bis eine Bedingung erfüllt ist. Die Bedingung kann ein Code oder einfach eine Regel basierende Bedingung sein.

Condition: es gibt die CodeCondition und die RuleConditionReference

CodeCondition: diese führt eine Methode der eigene code behind Klasse aus. Diese gibt `true` oder `false` zurück. Mit Code Condition sind Entwickler vertraut, hiermit lassen sich komplexe Bedingungen einfacher als in einer RuleConditionReference abbilden. Das Ergebnis einer CodeCondtion wird als Result Property der ConditionalEventArgs zurückgeliefert.

RuleConditionReference: diese kann extern als XML gespeichert werden, die Extension ist `.rules`. Das hat den Vorteil, dass dies extern durch andere Tools geändert werden kann, auch zur Laufzeit.

Parameterübergabe in Workflows: Bei einer Activity gibt es normalerweise einen Zusammenhang zu den Bewegungsdaten, also zum Bespiel zur Customer Id. Zum Unterschied zum Aufruf von Klassen in .net, wo Parameter bei Methoden übergeben werden, gibt es bei der Parameterübergabe in Workflows ein Dictionary von Name/Values.

1.1.3 Für welchen Workflow soll ich mich entscheiden?

Ein State Workflow wartet auf externe Events bevor er in einen anderen Step springt.

Ein Sequential Workflow ist ein zusammenhängender Fluss von Activities. Activities in einem Sequential Workflow warten nicht auf externe Instanzen um dann in den nächsten Step zu springen.

Man kann jeden State Workflow auch als Sequential Workflow abbilden. Früher order später wird dieser aber unüberschaubar werden.

2. Workflow Speicherformat

XOML *(Extensible Object Markup Language)* ist eine auf XML basierende Syntax. Ein Workflow kann als XOML gespeichert werden.

2.1 SqlPersistenceService

Per default laufen alle Workflows im Memory, solange auch die Applikation läuft, die den Workflow hostet. Zum Persistieren jeder einzelner Instanz des laufenden Workflows kann das mitgelieferte SqlPersistanceService verwendet werden. Damit wird der Zwischenstand der Workflows gespeichert, die Host Applikation kann jederzeit beendet und neu gestartet werden, außerdem können Workflows aus dem Speicher entladen werden wenn sie idle gehen. (Spart Memory, Performance)

Der Workflow kann damit am Filesystem oder in der Datenbank (SQL Server, SQL Express, …) gespeichert werden.

```
// Add the SqlWorkflowPersistenceService service
WorkflowPersistenceService persistenceService =
  new SqlWorkflowPersistenceService(
    "Initial Catalog=SqlPersistenceService;" +
    "Data Source=localhost;" +
    "Integrated Security=SSPI;",
    true,
    TimeSpan.FromHours(1.0),
    TimeSpan.FromSeconds(5.0));
workflowRuntime.AddService(persistenceService);
```

Parameter

ConnectionString: Verbindung zur bestehenden Persist DB

UnloadOnIdle: muss auf true stehen, sonst wird nichts persistiert. Gibt an, dass der Workflow entladen wird, wenn er idle ist. Default Wert false.

LoadingInterval: Zeitintervall, in der die Persist DB gepollt wird, um die persist records zu schreiben.

DB Installationsscript:

`c:\WINDOWS\Microsoft.NET\Framework\v3.0\Windows Workflow Foundation\SQL\EN\`

Tipp: SqlPersistanzService DB und Tracking DB können einzeln und getrennt verwendet werden. Wenn beides verwendet wird, empfiehlt sich aber beides in eine DB (SQL Server 2000, SQL Server 2005, oder SQL Server 2005 Express Edition) zu geben.

Microsoft Distributed Transaction Coordinator Service muss gestartet werden. Bei Verwendung eines Remote Servers muss Port 135 für Microsoft Distributed Transaction Center (MSDTC) offen sein.

2.2 Tracking

Da Workflows im Hintergrund laufen und die Workflow an sich auch komplexe Ausmaße annehmen werden, können diese mittels Tracking visualisiert werden. Bei der Instanziierung des Workflows wird das SqlTracking Service angegeben. Somit wird jede Instanz der Workflows getrackt.

Das Tracking ist für laufende als auch für beendete Workflows möglich, d.h. ich kann mir während des Ausführens als auch im Nachhinein den durchlaufenen Workflow ansehen. Die Darstellung erfolgt graphisch, der durchlaufene Workflow wird abgehackt dargestellt, der aktuellen Step mit einem grünen Dreieck.

DB Installationsscript liegt im Installationsverzeichnis:

`c:\WINDOWS\Microsoft.NET\Framework\v3.0\Windows Workflow Foundation\SQL\EN\`

3. Praxis

3.1 Workflow Hosting

Beispielanwendungen

- Automatisierungen von Abläufen
- Dokumenten Management
- Ticket Systeme
- Seitennavigation
- Komplexe Business Logic
- Häufige Änderungen an der Business Logic

Das Hosting kann in einem Service, über Webservice, asp.net, … erfolgen.

In diesem Beispiel in einem Windows Service:

```
#region fields
  WorkflowRuntime _wfruntime = new WorkflowRuntime();
  static AutoResetEvent _waitHandle = new AutoResetEvent(false);
  private System.Collections.ObjectModel.ReadOnlyCollection<WorkflowInstance>
    _loadedWorkflows;
#endregion
protected override void OnStart(string[] args)
{ ThreadPool.QueueUserWorkItem(new WaitCallback(RunWorkflow), args);
  _wfruntime.WorkflowCompleted +=
    new EventHandler<WorkflowCompletedEventArgs>(OnWorkflowCompleted);
}
private void RunWorkflow(object obj) {
  try
  {
    // Load WF from SqlWorkflowPersistenceService // Create the WorkflowRuntime
    using (WorkflowRuntime workflowRuntime = new WorkflowRuntime())
    {
       // Add the SqlWorkflowPersistenceService service
       WorkflowPersistenceService persistenceService =
         new SqlWorkflowPersistenceService(
           "Initial Catalog=SqlPersistenceService;" +
          "Data Source=localhost;Integrated Security=SSPI;",
           true, TimeSpan.FromHours(1.0), TimeSpan.FromSeconds(5.0));
       workflowRuntime.AddService(persistenceService);
       // Tracking
      workflowRuntime.AddService(
        new SqlTrackingService("Initial Catalog=Tracking;Data Source=localhost;" +
          Integrated Security=SSPI;"));
      // Set up the WorkflowRuntime event handlers
      workflowRuntime.WorkflowCompleted += OnWorkflowCompleted;
      workflowRuntime.WorkflowIdled += OnWorkflowIdled;
      workflowRuntime.WorkflowPersisted += OnWorkflowPersisted;
      workflowRuntime.WorkflowUnloaded += OnWorkflowUnloaded;
      workflowRuntime.WorkflowLoaded += OnWorkflowLoaded;
      workflowRuntime.WorkflowTerminated += OnWorkflowTerminated;
      workflowRuntime.WorkflowAborted += OnWorkflowAborted;
      _loadedWorkflows = workflowRuntime.GetLoadedWorkflows();
      Console.WriteLine("Workflows loaded: ", _loadedWorkflows.Count);
    }
  }
  catch (Exception ex)
  {
    EventLog.WriteEntry("Hosting WF in an Windows Service", e.ToString(),
      EventLogEntryType.Information, 1000);
  }
}
```

3.2 Workflow Tracking

Das Tool wird in den Samples mitgeliefert.

Workflow Monitor Sample

http://msdn.microsoft.com/en-us/library/ms741723.aspx

Um dies nutzen zu können, sind folgende Schritte notwendig:

- SQL Tracking DB muss angelegt worden sein.
- Eigener Workflow ist design und compiliert, Workflow wurde gestartet
- Erzeugt Workflow Assembly entweder ins Bin Verzeichnis vom Workflow Monitor kopieren oder die Assembly in den GAC geben (`gacutil –i demo.dll`)
- Workflow Tracking starten, Tracking DB angeben

3.3 Praxis Beispiel

In jedem organisieren Unternehmen gibt es wiederkehrende Abläufe, die man in einem Workflow Prozess abbilden kann. Dies betrifft einfache und komplizierte Vorgänge. Alleine durch die Analyse innerhalb einer Firma ergeben sich oft Schwachstellen, Lücken oder Missverständnisse die Arbeitsvorgänge undurchsichtig, unkontrollierbar und zeitaufwendig gestalten.

Es bietet sich an, dies als elektronischen Workflow abzubilden.

Anhand eines einfachen Beispiels möchte ich dies veranschaulichen.

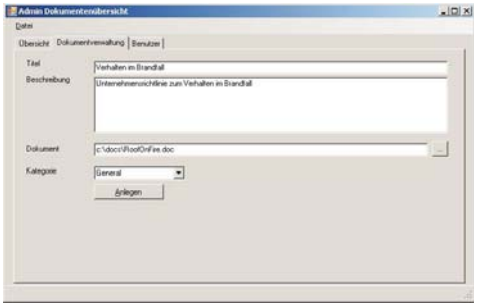
Es handelt sich dabei um einen Genehmigungsprozess, bei dem ein bereitgestelltes Dokument durch einen Gruppe oder einen Benutzer akzeptiert werden muss.

Ablauf

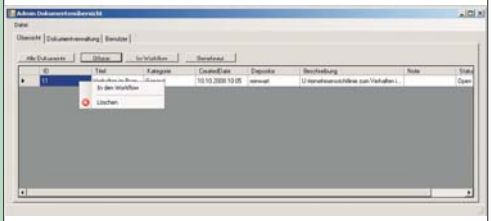
Eine leitende Person erstellte Dokumente und stellt diese in den Workflow. In dem Workflow sind drei Abteilungen definiert: IT User, System User, und allgemeine Benutzer.

Im Workflow sind diese Gruppen durch eine IfElseActivity aufgeteilt, daher kann jede Gruppe eine besondere Businnes Logic (BL) aufweisen. Eine solche BL lässt sich in einer CodeActivity unterbringen. In einer solchen CodeActivity lässt sich eigener Code ausführen.

Die Nutzung des zuvor erstellten Workflows ist in diesem Fall eine simple .net Windows Forms Anwendung, ist aber vom Workflow völlig unabhängig.



Ein Dokument mit den Attributen Titel, Beschreibung und Kategorie wird angelegt.



In der Übersichtsliste erscheint das Dokument und kann in den Workflow übernommen werden.

Technischer Teil

In der Applikation wurde das Persistence Service und das TrackingService der Windows Workflow Foundation eingebunden.

Persistence Service

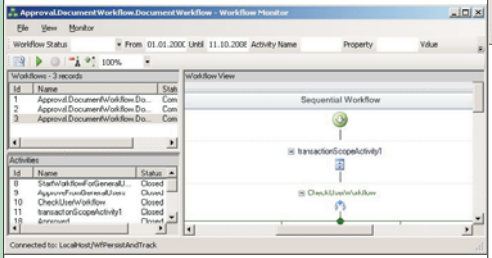
Jede Workflow Änderung eines Status wird in die Datenbank geschrieben.

TrackingService

Bietet die Möglichkeit, bei Instanz eines Workflow den aktuellen Status zu erfassen

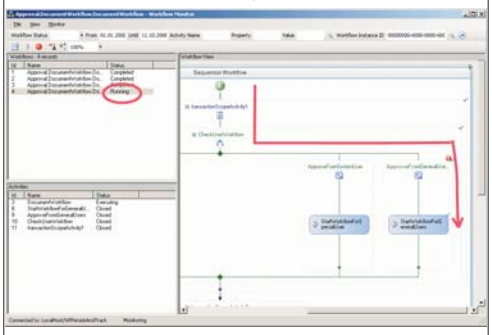
Tracking Monitor

Zeigt den aktuellen Workflow an ….

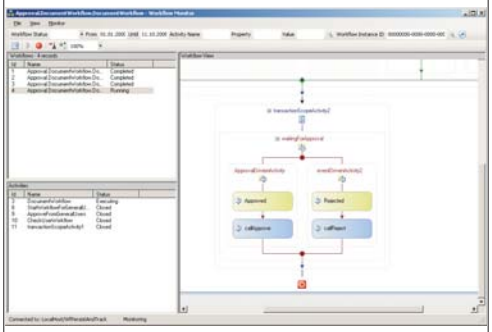


Damit lässt sich jeder aktuelle Step jedes einzelnen Workflow Überwachen und verständlich darstellen.

Das Dokument wurde in der Workflow gestellt, der Workflow läuft. *(workflow running)*

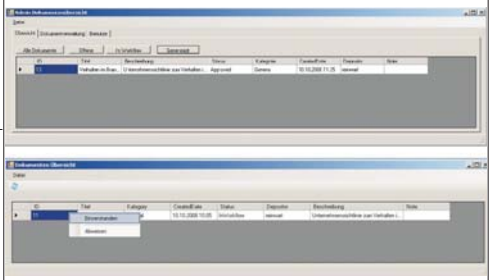


Der Workflow hat nun den Weg für die Kategorie General User genommen und wartet nun auf die Bestätigung durch den zugewiesenen User.



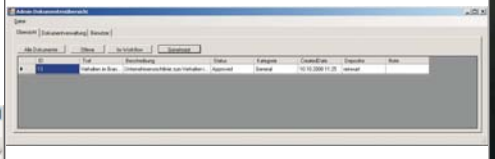
Der User hat nun die Möglichkeit, das Dokument, das nun diesen Workflow durchläuft zu bestätigen oder abzuweisen.

Für den User gibt es eine andere Applikation. Zu Beginn muss er sich authentifizieren, nach der Anmeldung sieht er die für ihn bestimmten Dokumente, für die eine Zustimmung notwendig ist.



Er liest sich den Inhalt des Dokuments durch und ist damit einverstanden.

Die Anwendung aus der Sicht des Administrators: Das Dokument wurde genehmigt und ist als *„Approved"* in der Liste geführt.



Der Administrator hat laufend die Übersicht, welche User die Dokumente bereits akzeptiert haben. Bei einer Überschreitung eines Zeitraumes könnte man in diesem Demo Workflow ohne viel Aufwand auch um eine Eskalationsstufe (z.B. automatische generiertes Email) erweitern.

4. Einschränkungen mit WF 3.0

Performance: Microsoft selbst dokumentiert dies in dem Dokument "Performance Characteristics of Windows Workflow Foundation".

(http://msdn.microsoft.com/en-us/library/aa973808.aspx)

Versionierung von Workflows

Ein Workflow kann jederzeit geändert werden, aber die bestehenden Workflows kann man nicht mit der neuen Definition weiterlaufen lassen. Wenn Sie bei einem Prozess plötzlich am Ende noch einen Schritt brauchen, können Sie diesen also nicht bei Workflows anwenden, die bereits gestartet sind (auch wenn diese langlebig sind).

Keine Kompatibilität zu SSIS-Workflows

Microsoft SQL Server Integration Services (SSIS) bietet eine ähnliche Umgebung wie WF, ist jedoch in keinster Weise kompatibel zu WF. Das Entwicklungsteam von SSIS hat sich aktiv gegen die Verwendung von WF als Basis ausgesprochen (einerseits war SSIS vor WF auf dem Markt, andererseits ist SSIS mehr auf Massendatenverarbeitung fokussiert, dafür wäre WF zu langsam).

Designer Control Hosting

Eigener Code zusätzlich zum WF Control notwendig, für einen WF Beginner nicht einfach. Mit dem Designer Control der Version 4.0 wird es einfacher sein.

5. Zukünftige Version WF 4.0

Auf der PDC 2008 wurde von Kenny Wolf, einem Microsoft Architect von WF und WCF, die Zukunft von WF in der Version 4.0 präsentiert. Workflow 4.0 soll zusammen mit dem .net Framework 4.0 und Visual Studio 2010 auf den Markt kommen.

In der Vorführung erkennt man die neue einfachere Oberfläche für das Design eines Workflow Prozesses. Aufgrund der Erkenntnis, das die WF Versionen 3.0 und 3.5 Schwächen im Bereich Funktionsumfang, Bedienung, Komplexität und Leistung aufweisen und die Kundenwünsche damit nicht immer befriedigt werden können, wurde WF von Grund auf neu geschrieben. Das Konzept der Aktivitäten im Workflow und die Persistierung und die Möglichkeit von Monitoring bleibt. Es soll eine 10 bis 100-fache Leistungssteigerung geben, ebenso eine volle Kontrolle über die Persistierung. WF 4.0 wird rein deklarativ sein.